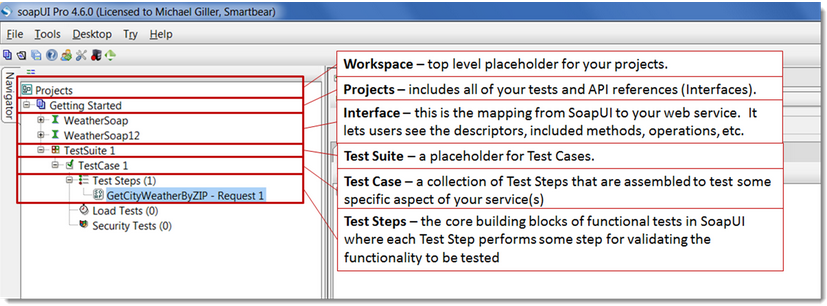
# Soap UI

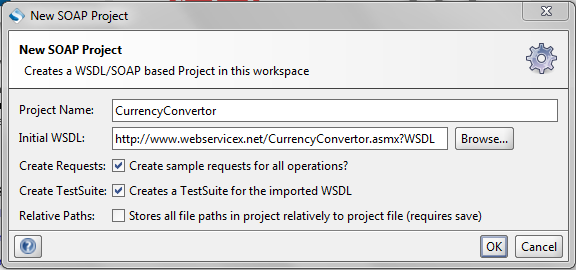
Structure:



## Create Sample Project

1. In soap UI > File > New SOAP Project.
2. Load WSDL,

WSDL Sample: <http://www.webservicex.net/CurrencyConvertor.asmx?WSDL>

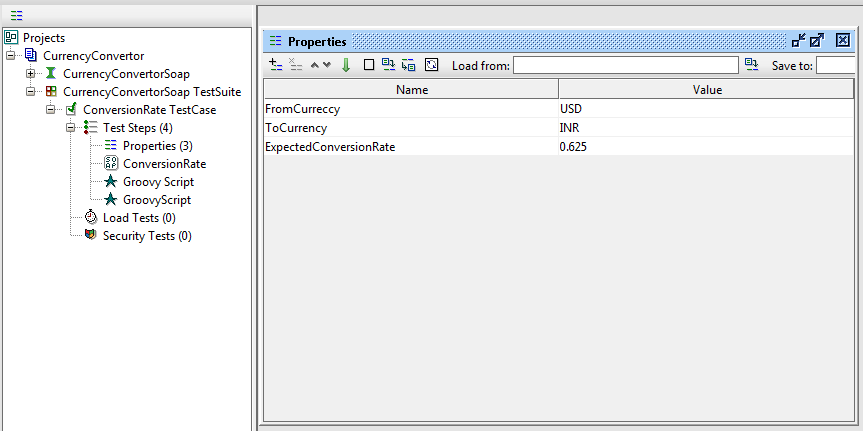


1. Click OK…
2. WSDL for many Web Services are available at <http://www.webservicex.net/ws/default.aspx>,
3. Open WSDL in browser for confirmation (Availability).

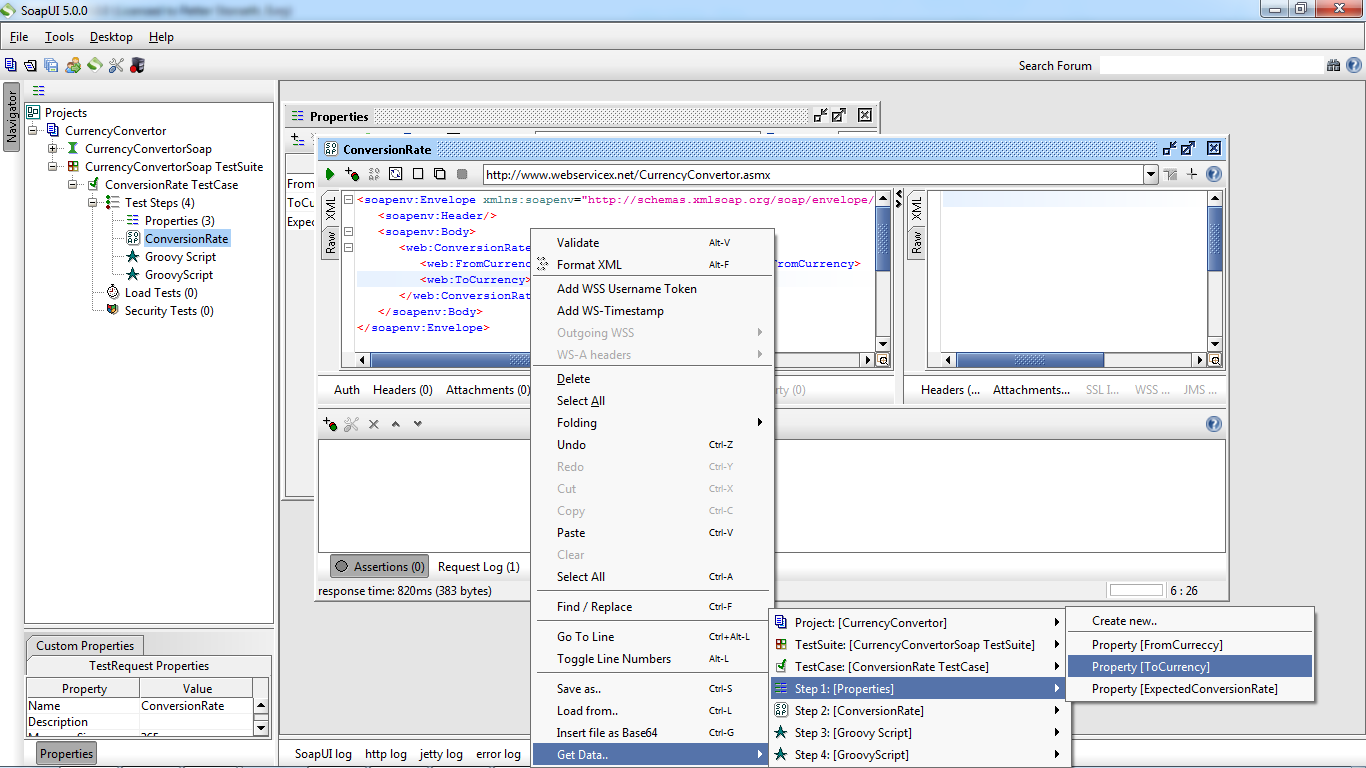
## Add Test Steps

* Right click on test step
* Click “Add Step” OR “Insert Step”
* Select the test Step to be added.

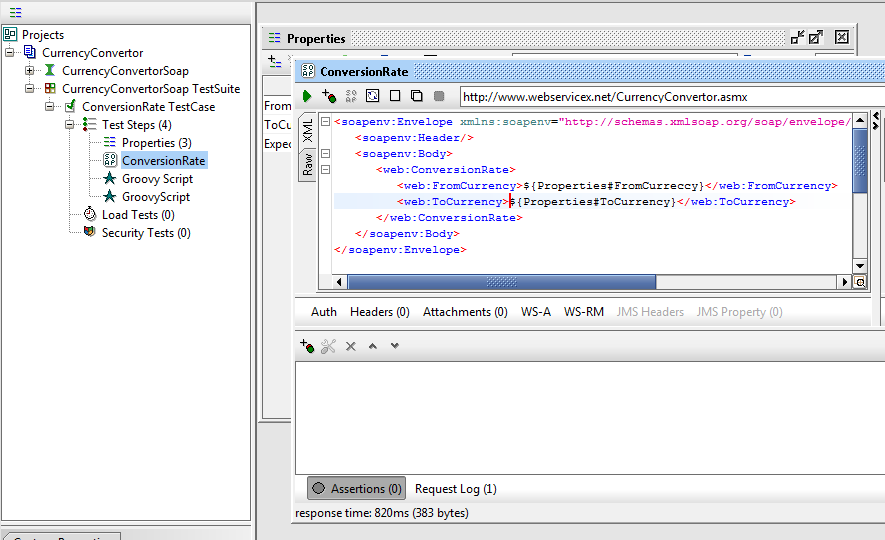
1. Use a property test step as input data to **test request.**



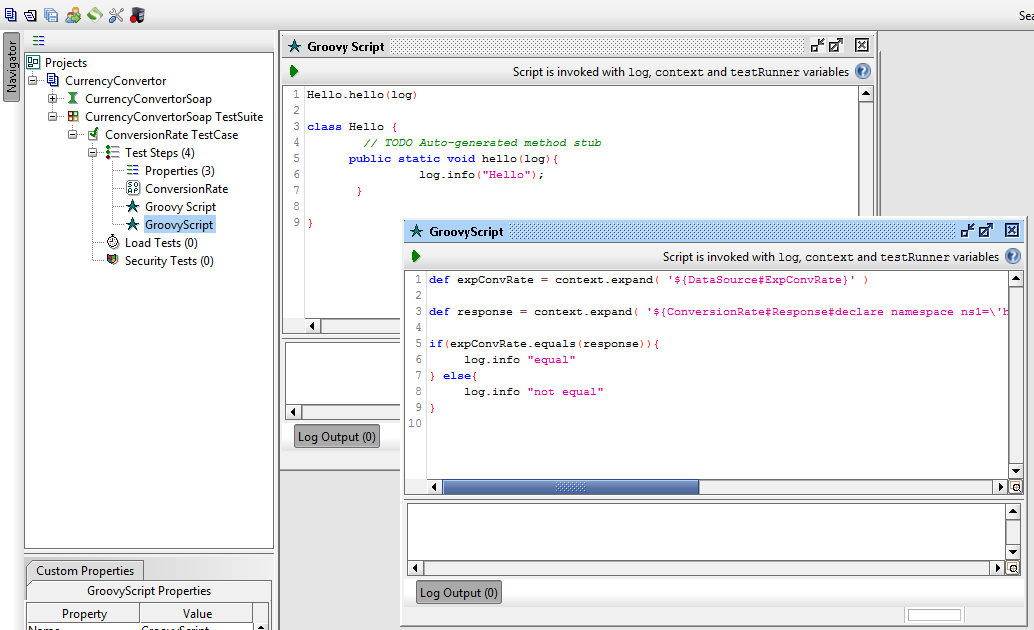
1. Configure the same in test request (ConversionRate) as below (Right click > Get Data> Test Step > property).



Configured inputs:

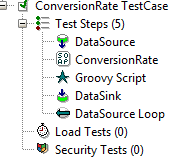


1. Add a groovy script test step and run some scrip.



## Validation using Groovy Script

* Add Test Step Groovy Script:



Add Script:

//declare Groovy Util context

def groovyUtils = new com.eviware.soapui.support.GroovyUtils( context )

//declare variable for Expected Rate

def expectedRate = context.expand( '${DataSource#ExpectedRate}' )

//declare response Xml holder

def holder = groovyUtils.getXmlHolder("ConversionRate#Response")

//Get value From Response XML using XPATH

def actRate = holder.getNodeValues("//\*:ConversionRateResult")

//To write Data to data sink (variable holding path to data sink)

def runnerComplete = testRunner.testCase.testSteps.DataSink

if (expectedRate.equals(actRate)){

TestStatus = "Pass"

} else {

TestStatus = "Fail"

}

//write result into Data Sink

runnerComplete.setPropertyValue("TestStatus",TestStatus)

## Request manipulation

//declare Groovy Util context

def groovyUtils = new com.eviware.soapui.support.GroovyUtils( context )

//declare request Xml holder

def holder = groovyUtils.getXmlHolder("ConversionRate#Request")

// declare variables from Data Source

def fromCurrency = context.expand( '${DataSource#FromCurrency}' )

def toCurrency = context.expand( '${DataSource#ToCurrency}' )

//update variable values into request xml holder

holder.setNodeValue("//\*:FromCurrency",fromCurrency)

holder.setNodeValue("//\*:ToCurrency",toCurrency)

holder.updateProperty()

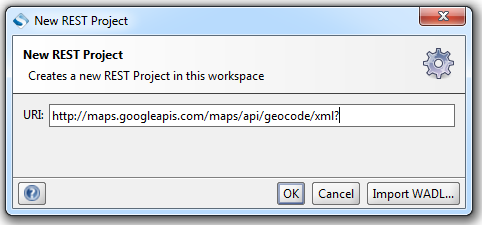
**For Further details on Groovy Script refer,**

<http://www.soapui.org/Scripting-Properties/tips-a-tricks.html>

# Rest Services

## Creating a sample rest project:

1. In soap UI > File > New REST Project.
2. Provide the URL: (can include parameters in URL or without parameter)



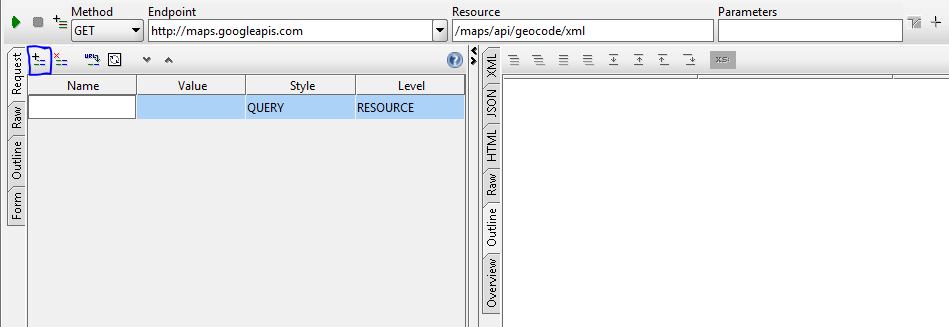
URL with parameters:

<http://maps.googleapis.com/maps/api/geocode/xml?address=1600+Amphitheatre+Parkway,+Mountain+View,+CA&sensor=false>

URL without parameters:

<http://maps.googleapis.com/maps/api/geocode/xml>?

1. Adding parameters to request:



1. Create test suite
2. Add groovy script to build a JSON object:

//define a JSON builder

def builder = new groovy.json.JsonBuilder()

//declare test request (used to set JSON content to request)

def request = testRunner.testCase.testSteps.Request1.getTestRequest()

//declare variables from data source to be used in JSON request

def FromCurrency = context.expand( '${DataSource#FromCurrency}' )

def ToCurrency = context.expand( '${DataSource#ToCurrency}' )

// log.info prints the variable value

log.info FromCurrency

//Build the Request

builder{

// condition if parameter is optional input field (declared in JSON object only if it has value)

If (FromCurrency != “”) {

fromCurrency FromCurrency

}

toCurrency ToCurrency

}

//print the JSON object into request body.

request.setRequestContent(builder.toPrettyString())

1. Response validation is same as soap response validation, only difference is while declaring XML holder for response.

//declare response Xml holder

def holder = groovyUtils.getXmlHolder("ConversionRate#ResponseAsXML")